DOCUMENT RESUME

ED 315 034 IR 014 136

AUTHOR Chandra, Peter
TITLE Adaptable Interactive CBL Design Tools for Education.
PUB DATE Jul 88
NOTE 11p.; Paper presented at the European Conference on Computers in Education (ECCE88) (Switzerland, July 24-29, 1988).
PUB TYPE Reports - Descriptive (141) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Authoring Aids (Programing); *Computer Assisted Instruction; *Courseware; *Formative Evaluation; *Instructional Design; *Instructional Development; *Media Adaptation; Models

ABSTRACT

The design team approach to the development of computer based learning (CBL) courseware relies heavily on the effective communication between different members of the team, including up-to-date paperwork and documentation. This is important for the accurate and efficient overall coordination of the courseware design, and for future maintenance of the courseware developed. These require a greater investment in time and finance, resources that the field of education can hardly afford. General purpose application packages (or content-free software) could be adapted so as to provide interactive tools for CBL design, particularly in the areas of design specification and storyboarding, flowcharts, user documentation, and initial courseware evaluation analysis. This is especially helpful for the CBL designer/author who does not know, or want to be involved in the detailed programming, or who does not have access to development tool kits or powerful workstations. Some specific examples are given using readily available general purpose software packages developed for microcomputers, especially for the Macintosh microcomputer. (8 references) (Author/GL)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED315034

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.
Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.
"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Peter Chandra
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

(Paper to be presented at the European Conference on Computers in Education (ECCE88), Switzerland, 2 Jul 1988)

ADAPTABLE INTERACTIVE CBL DESIGN TOOLS FOR EDUCATION

Dr. Peter Chandra

Computer Assisted Learning Group, Computing Unit,
University of Surrey, Guildford, Surrey, GU2 5XH, England.

The design team approach to the development of Computer Based Learning (CBL) courseware relies heavily on the effective communication between different members of the team, including up-to-date paperwork and documentation. This is important for the accurate and efficient overall coordination of the courseware design, and for future maintenance of the courseware developed. These require a greater investment in time and finance, resources that the field of education can hardly afford. General purpose application packages (or content-free software) could be adapted so as to provide interactive tools for CBL design, particularly in the areas of design specification and storyboarding, flowcharts, user documentation, and initial courseware evaluation analysis. This is especially helpful for the CBL designer/author who does not know, or want to be involved in the detailed programming, or who does not have access to development tool kits or powerful workstations. In this paper, some specific examples are given using readily-available general purpose software packages developed for microcomputers, especially for the Macintosh microcomputer.

1. INTRODUCTION

One of the approaches in developing good Computer Based Learning/Training courseware is the Design Team Approach [1,2,3]. Such a model is shown in Figure 1 which depicts the different expertise provided by the team members for the various design stages [4]. The design team approach, however, has been hampered by the lack of suitable authors who have the technical appreciation of how their ideas and teaching strategies can be implemented on a computer. In the development of Computer Based Learning (CBL) courseware, it would be useful if design tools were provided for the author/designer to be able to place his ideas in computer form so as to obtain an immediate visual feedback. This is particularly helpful if the author does not want to be involved in the programming of the courseware, but wants to ensure that the final outcome meets his requirements without the need for very detailed specifications or reliance on the programmer's interpretation of an author's brief specification - hence, one of the reasons for the increasing popularity of authoring systems or development tool kits. However, most authoring systems and tool kits still require a certain amount of programming skills which most subject authors do not possess [5,6]. In addition, most subject experts in the primary and secondary education level have access only to microcomputers (if any at all) and not minicomputers or workstations, and so are not in possession of powerful development systems or tool kits on which to formulate their ideas directly on to the computer screen. There is also the problem of making sure that any design documentation is kept up-to-date with both the designers and programmers keeping well informed of the latest changes made.

Adapting existing content-free software can be advantageous to designing CBL materials, especially when the designer or his educational establishment does not have the resources or finance to develop appropriate design tools to meet the designer's specific needs. The advent of system software that provides concurrent running of software in an environment where data can be transferred between software packages has further stimulated such ideas. This

IR014136

paper provides some suggestions of how generally available application software can be adapted or used as interactive design tools for programming-shy designers, following the design model shown in Figure 1. Although the examples given are based on software designed for the Macintosh microcomputer, the principles, ideas and suggestions written here could be implemented on software for any microcomputer. In this paper, the word "author" will refer to the writer of this paper, and the word "designer" to the author and/or designer of an intended CBL courseware.

2. DESIGN SPECIFICATIONS AND STORYBOARDS

It would be helpful to the designer at the initial design stage (Stage 1 in Figure 1) to be able to implement different ideas directly on the computer, so as to see and "get a feel" of the effect of his ideas on the computer screen - what the author calls "interactive design". It is a level in-between paper-based design and rapid-prototyping. This is where the designer would be able to formulate his ideas directly on to the computer and, with the aid of some software application programs, provide an immediate visual feedback of his ideas. In addition, any instruction that was needed to be communicated to the computer programmer could be made on-screen, with additional notes to himself or other evaluators.

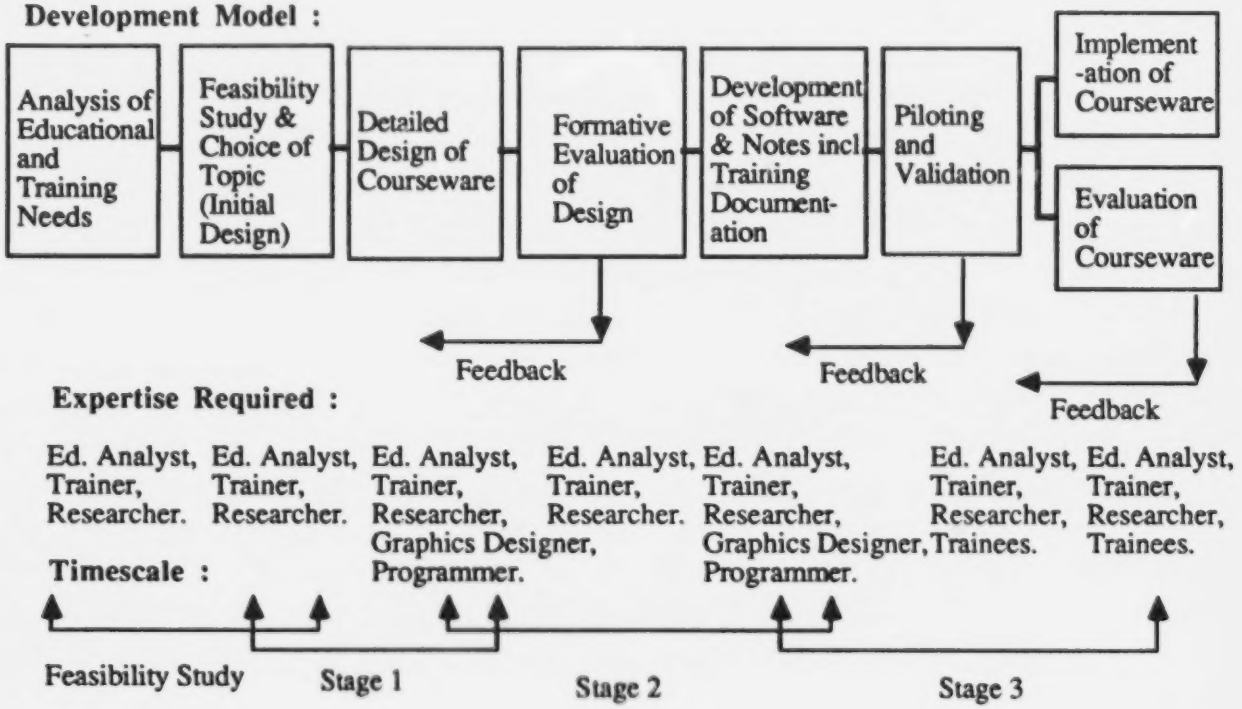

Figure 1. The Team Approach to CBL Courseware Design

As an example, this author uses two general software packages - a graphics editor (MacPaint) and a sequencer (Slide Show Magician) to accomplish such an effect. Figure 2 shows the Design Specification Sheet (DSS) layout that was designed to provide the template for this interactive design storyboard. It is, in effect, a graphics file, i.e. a MacPaint document. A box is provided to demarcate the screen design, with two boxes provided for instructions to the programmer and for the designer's notes. The designer first draws out the intended screens (with the graphics editor) and saves it as separate files. These become his design specification sheets (Figure 3 provides an example). He then uses the sequencer (Slide Show Magician) to link in the separate graphic files according to the sequence in which he thinks he would like the program he is designing to be presented. Added facilities of interaction and routing are provided by this sequencer. For example, the designer is able to control the time delay between each screen/design specification sheet display, and to route it according to screen location pointers. With the sequencer, the designer is thus able to obtain an immediate "feel" of how the intended courseware should be presented.

Using a third program, which acts at the system level, to provide the running of programs concurrently or to have different programs co-resident (in this case, the Macintosh program is called Switcher), the designer is able to toggle between the graphics program and the sequencer program. The designer is thus able to make any changes he sees necessary on the design specification sheet, and by toggling on to the sequencer program, he is able to "run his program" and obtain immediate feedback on the flow and execution of his program. In this way, the designer can interactively design his specification sheets. Once satisfied, a print-out can be obtained providing the necessary design documentation for the programmer to implement on a computer, a list of the intended sequence, and further notes for the designer. Interactive and automated design specification sheets are thus achieved.

3. STRUCTURE/FLOWCHART GENERATOR

An important documentation needed by designers and programmers at the initial and detailed design stages of the development process (Stages 1 and 2 in Figure 1), is the flowchart or structure chart. For educational or training courseware, where the level of routing required is basic and straightforward, the possession of an outline program (outliner) that has the facility of producing tree diagrams could be adapted to provide an elementary structure chart generator. For example, an outline program called MORE (the Macintosh version of ThinkTank 512 for the IBM pc) has the ability, at a keystroke, to produce tree diagrams from the headings assigned in the outliner program (what they term as "tree charts"). So, for example, Figure 4 shows the headings, including different sub-levels of headings, that have been assigned as outlines within MORE. These, for our purposes, are the "procedures" for the intended software.

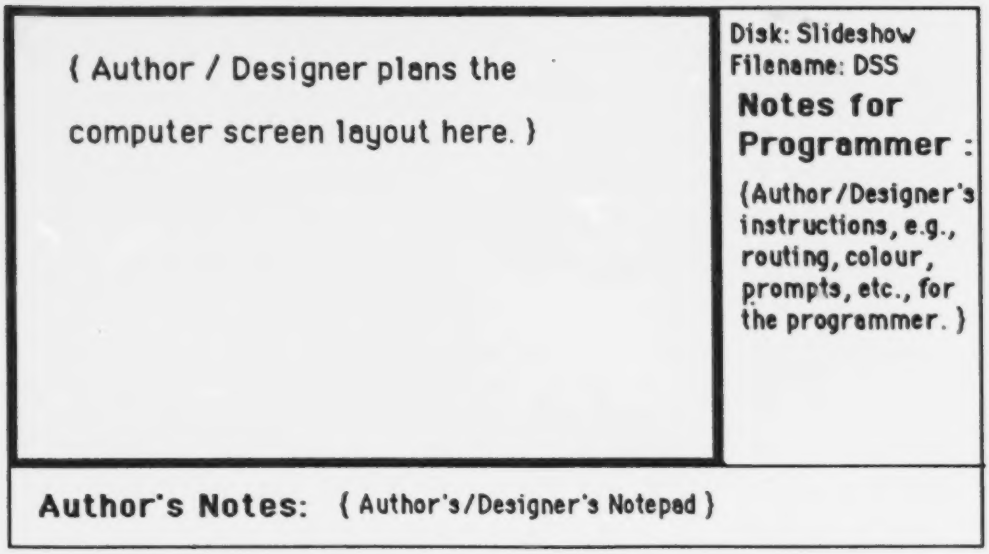
{ Author / Designer plans the
computer screen layout here. }
Disk: Slideshow
Filename: DSS
Notes for Programmer :
(Author/Designer's instructions, e.g., routing, colour, prompts, etc., for the programmer.)
Author's Notes: { Author's/Designer's Notepad }

Figure 2. Example of a Design Specification Sheet, DSS (a MacPaint Document)

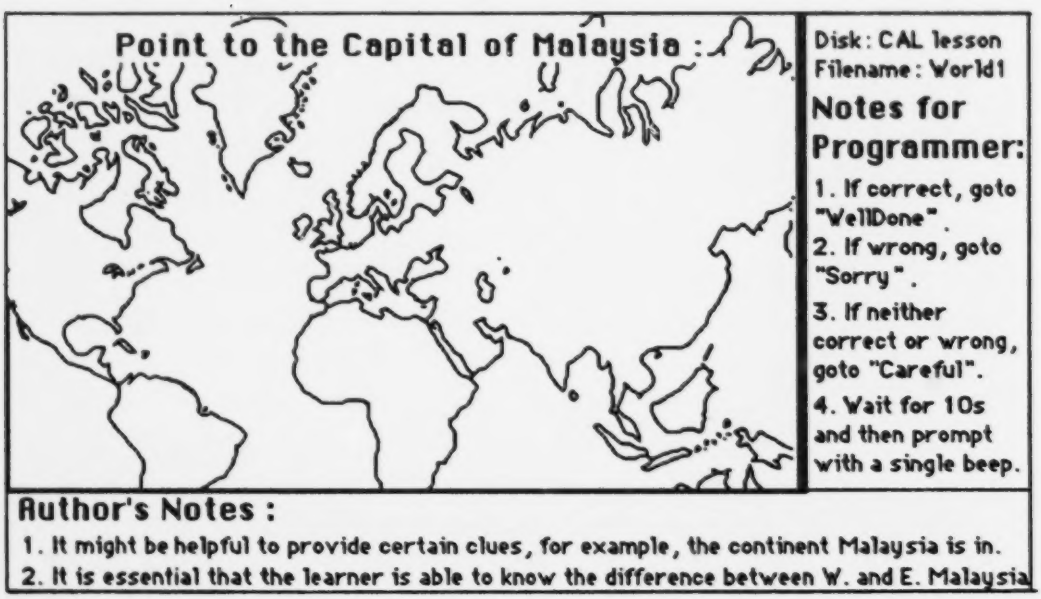
Point to the Capital of Malaysia :
Disk: CAL lesson
Filename: World1
Notes for Programmer:
1. If correct, goto "WellDone".
2. If wrong, goto "Sorry".
3. If neither correct or wrong, goto "Careful".
4. Wait for 10s and then prompt with a single beep.
Author's Notes :
1. It might be helpful to provide certain clues, for example, the continent Malaysia is in.
2. It is essential that the learner is able to know the difference between W. and E. Malaysia

Figure 3. Example of a DSS with author's instructions and notes

Intro. to Commonwealth Countries
1.Looking at these countries by Continents(*)
1.1. Introducing .... Asia (*)
1.1.1. Basic Information : Position, History
Malaysia (*)
Population (o)
Repeat until Done
Climate (o)
Repeat until Done
The Capital (o)
Repeat until Done
Question/Test (o)
Q1: Population Profile
Q2: Relief of country
Q3: Main towns
Q4: Profile of capital
Q5: Similarities/Differences of E/W Malaysia
Repeat until 4/5 questions correct
Thailand (*)
Population (o)
Repeat until Done
Climate (o)
Repeat until Done
The Capital (o)
Repeat until Done
Question/Test (o)
Q 1-5
Repeat until 4/5 correct
Look through any country again (*)
1.2. Introducing ...Europe (*)
2. According to Language (*)
2.1. Chinese
2.1.1. Origins
2.1.2. Type of script
2.2. English
2.2.1. Origins
2.2.2. Type of script

Figure 4. Headings and sub-headings in MORE

At the stroke of a control key, this set of headings and sub-headings can be made to become a tree chart, which in essence is the makings of a structure diagram (See Figure 5). Changes can be made to the outlines and a different tree diagram generated immediately. The use of such an outliner, with its sophisticated wordprocessing facilities and the added facility of a tree diagram generator, can thus provide a very handy interactive tool for the design of simple flow/structured charts.

4. INTERACTIVE USER DOCUMENTATION

Poor user documentation has led to a number of good software packages being under-utilised. In CBL courseware, it is important to distinguish further whether the "user" is the trainee/pupil or the trainer/teacher (Stage 3 in Figure 1). Appropriate, and thus different "user" documentation, should be written for the different user types (between trainee and trainer, and even between different trainees) if proper implementation and integration of the CBL courseware is to take place [7].

Increasingly, a greater number of user documentation is now being produced on wordprocessors before being printed out. The author's rationale is that if such documentation is on a wordprocessor, why not use a general purpose hypertext program to generate an interactive user documentation? "Hypertext" was a term coined in the early 1960s to describe a non-linear body of information with "links" between documents or different parts of a document, that guides the reader from one to another [8]. At present, such a program (called GUIDE) exists for the Macintosh microcomputer which also provides all the facilities of a powerful wordprocessor.

The skills needed to write hypertext media, however, are not exactly the same as compared to those needed for wordprocessing. More thought has to be given in considering the amount of text to be revealed or hidden, but readily available. In this way, help prompts could be easily embedded to guide the user along the way, only when it is either necessary or called upon. Time will also have to be spent on what and how the writer would want his document to "open-out" and flow. The full power of graphic images and drawings of the Macintosh are also available. This is beneficial where graphic illustrations are crucial to explain better or describe a particular concept or procedure. Other facilities provided by such hypertext programs include the provision of "pop-up" text which could be used as calls for definitions of words, glossary/index, etc. (see Figure 6).

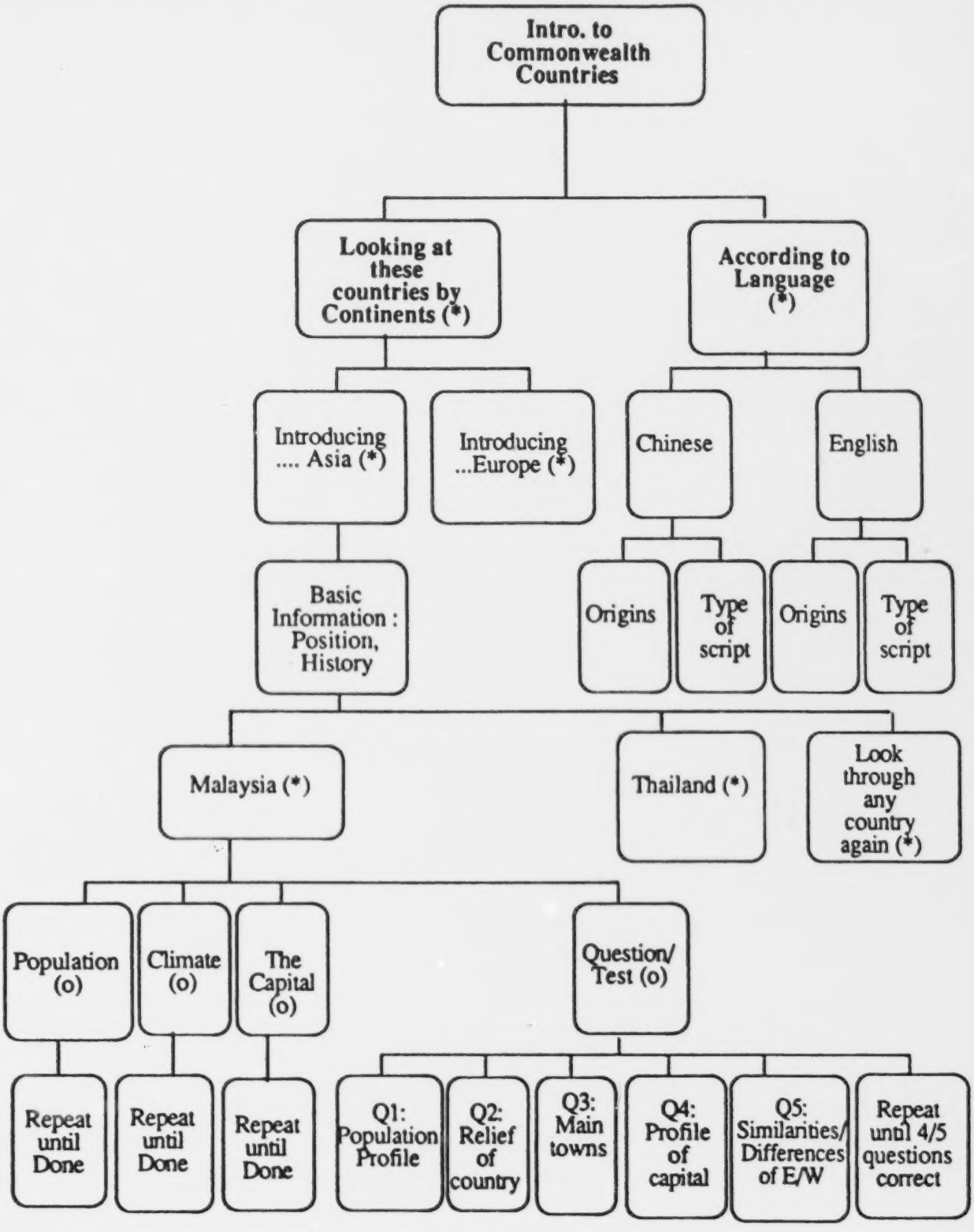

Figure 5. Tree Chart produced from MORE's Outlines

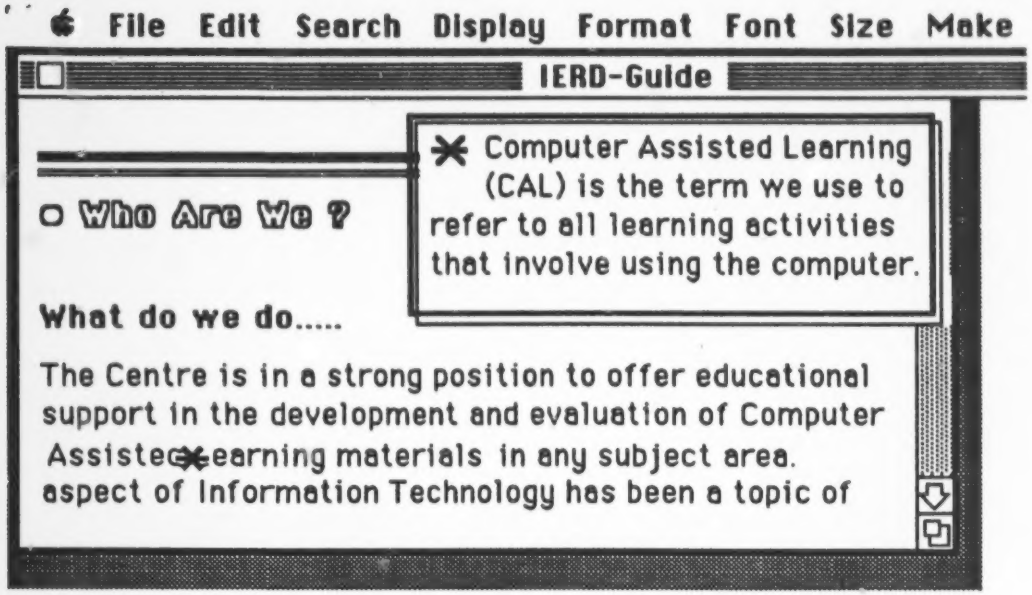
File Edit Search Display Format Font Size Make
IERD-Guide
Computer Assisted Learning (CAL) is the term we use to refer to all learning activities that involve using the computer.
o Who Are We ?
What do we do.....
The Centre is in a strong position to offer educational support in the development and evaluation of Computer Assisted Learning materials in any subject area.
aspect of Information Technology has been a topic of

Figure 6. Example of a Hypertext document with "pop-up" window

This "interactive documentation" could be taken one step further to include animated sequences to elaborate powerfully a difficult procedure. For example, using an animated simulation to show a naive user how a computer program is loaded into a microcomputer. The use of such a software application package, as for example, "Videoworks", would be suitable for this type of need. Videoworks is a package which allows different objects/drawings to be assigned as "actors". Such "actors" could be made to "play out" the sequence to illustrate the action required. Taking the above example, a sequence is animated to show the precise way in which the user needs to put a disc into the disc drive and enter the appropriate commands to load in the computer program.

This animation could be activated at the appropriate part of the interactive user documentation by using another general purpose program that generates "macros" (similar to that used in spreadsheets). An example of such a program for the Macintosh computer is called TEMPO. In this way, a specific user action could be suggested by the interactive documentation which involves the pressing of a specific key. This calls up a "macro", which then calls up the animated program and appropriate animated sequence, via the program Switcher. Once the dynamic illustration is over, another appropriate key stroke would bring the user back to the user documentation where he had left off.

It would not be difficult to see that such a combination of general application programs could also be used as an interactive and dynamic storyboard, where the implementation of the program would be on other types of computers. It could also be possible that these set of programs could be used for the actual delivery of courseware on the particular computer itself. This concept is an adjunct of hypertext, called "hypermedia". Hypermedia is the logical

extension of hypertext, encompassing graphics, animation, video, sound, etc. Blocks, or pieces of documents can thus be linked to form multi-webs of information which could be presented to the learner/student in an open-ended but dynamic way [8].

5. AUTOMATED EVALUATION ANALYSIS

It is essential that any CBL development project should have time built in to conduct an initial evaluation of the courseware being produced (end of Stage 3 in Figure 1). This phase of a project has a tendency to be excluded, especially when there is a pressure of deadlines to be met. The analysis of the evaluation might consist of a questionnaire which would provide simple indicators (involving simple frequency counts) of the performance of the courseware, plus a bank of relevant, but short, comments made. This would be sufficient to provide valuable, initial feedback. An automation of this stage would be useful.

One of the ways of achieving this is to provide the questionnaire directly on computer. It is suggested in this paper that this questionnaire is actually a template of a spreadsheet - the questions are the rows of the spreadsheet, and the answers are those to be placed in the columns (normally a rating scale of 1-5). The spreadsheet itself is part of an integrated package with database, chart generating and wordprocessing programs. The questionnaire is completed by the courseware evaluator directly on the computer. Once a number of questionnaires have been completed in this way, the results of each questionnaire (found in the columns of each spreadsheet file) can then be copied on to the columns of a master spreadsheet (with the use of suitable macros). As a questionnaire comes in, the master spreadsheet could be updated. The value of using an integrated package (for example, Framework II on the IBM pc, or Jazz on the Macintosh) is that the values of these columns could be mathematically manipulated with simple statistical functions provided, and the questionnaire results automatically displayed in the form of suitable charts or graphs.

A further idea would be to provide the questions for the questionnaire as records of a file in a database. A library of questions could thus be compiled. Combinations of questions could be found by the use of keywords assigned to specific questions in the database. With the use of specific macros in an integrated package, these questions could be automatically selected from a database and copied on to a spreadsheet which then becomes the template for your interactive questionnaire. In this way, customised questionnaires could be designed, depending on the type of courseware being evaluated and the target audience. For example, the questions directed for a teacher/trainer would be different to the questions needed for a pupil/trainee; while a CAL software could be evaluated and distinguished from a CAD program by using different questions.

6. CONCLUSION

The suggestions discussed in this paper can provide design tools which facilitate up-to-date and immediate feedback. Using general purpose software packages to provide immediate reflection of any changes made in the design, affords an interactive form of designing courseware. Furthermore, using design tools on a computer which generates the "paperwork" can be advantageous as it provides a rapid form of updating of documents. In this way, better record keeping of existing documents and a more efficient form of housekeeping can be established. These interactive design and user documents could be mailed electronically to other design team members, an option particularly helpful if the team is widely spread out geographically. Any printouts obtained would also be legible, an added advantage when it comes to different handwriting amongst team members !

It is tempting for the designer to seek for design tools that could generate actual software code to be implemented on different computers, i.e. design tools which provide machine-independent code which could be translated automatically for different computers. Maybe the day will come soon when this could be achieved at a price that educational establishments can afford. However, the value of the suggestions provided here in this paper is that these programs are available now for the educational market, and that by using pieces of existing general purpose software, designing CBL courseware could be easier and more enjoyable.

REFERENCES

[1] Beech, G. (1983) Computer Based Learning, Sigma.

[2] Crossfield, L.P. & Hinton, T. (1984) "A Methodology for the Design and Development of CAL Courseware". Paper presented at the Micros in Education Conference, Loughborough, April.

[3] Dean, C. and Whitlock, Q. (1983) A Handbook of CBT. Kogan Page, London.

[4] Chandra, P.; Black, T. and Hinton, T. (1986) "The Design of Interactive Video Training Courseware for Water and Sanitation Engineers in Peru - Some Initial Considerations". Proceedings of the Int. Conf. on Courseware Design and Evaluation, L. F. Lewis and B. Feinstein (eds), Israel, April 8-13, pp 393-398.

[5] Black, T. (1987) "Prototyping CAL Courseware: A Role for Computer-Shy Subject Experts" Paper presented at the Ed. Tech. Int. Conf. (ETIC-87), Univ. of Southampton, 13-15 April.

[6] Hinton, T. (1984) " Authoring Systems in CBT" Paper presented at the conference on CBT organised by the Danish Postgraduate Engineering Society, Denmark, 3 May.

[7] Laubli, M., Pope, M. and Hinton, T. (1985) "Implications for the Cross-Cultural Transfer of Computer Based Education Courseware - A Case Study". PLET 22(3), pp224-229.

[8] Osgood, D. (1987) "The Difference in Higher Education" in BYTE, Feb., pp165-178.